Generalized Response Surface Methodology

Kleijnen, J.P.C.

Publication date:
2006

CentER

Discussion Paper

No. 2006–77

GENERALIZED RESPONSE SURFACE METHODOLOGY: A NEW METAHEURISTIC

By Jack P.C. Kleijnen

August 2006

ISSN 0924-7815

TILBURG UNIVERSITY

manuscript No.
(will be inserted by the editor)

Generalized Response Surface Methodology

A new metaheuristic

Jack P.C. Kleijnen

Tilburg University, Postbox 90153, 5000 LE Tilburg, the Netherlands
e-mail: kleijnen@uvt.nl; phone 31-13-466-2029
website: http://center.uvt.nl/staff/kleijnen/

The date of receipt and acceptance will be inserted by the editor

Abstract Generalized Response Surface Methodology (GRSM) is a novel general-purpose metaheuristic based on Box and Wilson's Response Surface Methodology (RSM). Both GRSM and RSM estimate local gradients to search for the optimal solution. These gradients use local first-order polynomials. GRSM, however, uses these gradients to estimate a better search direction than the steepest ascent direction used by RSM. Moreover, GRSM allows multiple responses, selecting one response as goal and the other responses as constrained variables. Finally, these estimated gradients may be used to test whether the estimated solution is indeed optimal. The focus of this paper is optimization of simulated systems.

Key words experimental design - multivariate regression analysis - least squares - Karush-Kuhn-Tucker conditions - bootstrap
JEL: C0, C1, C9

1 Introduction

The importance of *optimizing* engineered systems (man-made artefacts) is emphasized in the 2006 NSF panel reported in Oden (2006). That report also emphasizes the crucial role of *simulation* in engineering science. An essential characteristic of simulation—compared with Mathematical Programming (MP)—is that in simulation the objective function (which is the function to be minimized or maximized) is not known explicitly; actually, this function is defined implicitly by the simulation model (computer code, computer program). The simplest optimization problem has no constraints on the inputs or outputs of the system (either real or simulated). The simulation model may be either deterministic or random (stochastic, discrete event). In

this article, however, I will focus on random simulation, which (by definition) uses Pseudo-Random Numbers (PRNs)—PRN are defined as independently and uniformly distributed on the interval $[0, 1]$. Nevertheless, I expect that the methodology for optimization of random simulation models—known as *simulation optimization*—is also relevant for optimization of deterministic simulation models and real-world systems.

The *simplest optimization* concerns the minimization of the expected value of a single (univariate) simulation output. An academic example is an (s, Q) inventory management simulation aimed at minimization of the expected value of the total inventory costs (the sum of inventory carrying, reorder, and out-of-stock costs), and the decision variables are the reorder level s and the order quantity Q. (Implicit input constraints are that these two decision variables must be nonnegative.) More complicated examples are inventory-production simulation models in logistics and operations management. Furthermore, telecommunications often uses queueing simulation models that try to minimize a given deterministic function of either the expected customer delay or the server blocking probability (such a probability may be formulated as the expected value of a binary variable).

In practice, however, simulation models have *multiple outputs*. Examples are many practical inventory models that require the inventory system to satisfy a minimum service rate (or fill rate), because the out-of-stock costs are hard to quantify (an example is Ivanescu et al. 2006). I shall formalize this type of problems in Section 3.

Metaheuristics can be used to optimize a simulated system. These methods treat the simulation model as a 'black box'; i.e., they observe only the inputs and outputs of the simulation model. In this article, I focus on *Response Surface Methodology* (RSM). This method is often ignored in the literature on metaheuristics. Nevertheless, RSM is a method that is often applied in real-life experiments; see Section 2 and also the Design-Expert software (see www.statease.com). (Some authors outside random simulation speak of RSM, but mean 'what-if' analysis—not sequential or iterative optimization; see, for example, Downing et al., (1985) and Olivi (1984).)

Generalized Response Surface Methodology (GRSM) is a novel general-purpose metaheuristic based on Box and Wilson (1951)'s classic RSM. Both GRSM and RSM estimate local gradients to search for the optimal solution. These gradient estimates use locally fitted first-order polynomials. GRSM, however, uses these gradients to estimate a better search direction than the steepest ascent direction used by RSM. Moreover, GRSM allows multiple responses, selecting one response as goal and the other responses as constrained variables. The input variables may also be subjected to box constraints. Finally, GRSM uses the estimated gradients to test whether the estimated solution is indeed optimal.

In this article, I focuses on *expensive* simulation; i.e, it takes much computer time to compute a single realization of the time path of the simulated system. For example, 36 to 160 hours of computer time were needed to simulate a crash model at Ford Motor Company; see the panel discussion on

optimization in Simpson et al. (2004). This panel also reports the example of a (so-called 'cooling') problem with twelve inputs (decision variables, factors), ten constraints, and one objective function. For such expensive simulations, many simulation optimization methods are unpractical; for example, OptQuest (an add-on to random simulation software products such as Arena and Simul8) requires relatively many simulation replicates and input combinations (points, scenarios); see Kleijnen and Wan (2006).

In this article, I summarize previous work that I did on classic RSM, and recent work on GRSM. That recent work is scattered over several papers, referenced in the following sections. More details are given in Kleijnen (2007).

The remainder of this article is organized as follows. Section 2 summarizes classic RSM, and the Adapted Steepest Ascent (ASA) search direction developed by Kleijnen et al. (2004). Section 3 summarizes GRSM for simulation with multiple responses, developed by Angün et al. (2006). Section 4 summarizes a procedure for testing whether an estimated optimum is truly optimal—using the Karush-Kuhn-Tucker (KKT) conditions—developed by Bettonvil et al. (2006). Section 5 presents conclusions. As many as 34 references enable the readers to study further details.

2 Classic RSM: single output and no constraints

Box and Wilson (1951) 's classic RSM searches for the combination of quantitative inputs that minimizes the univariate output of a *real-world* system (or maximizes that output: simply add a minus sign to the output). Numerous applications are given in Myers et al. (1989)'s excellent survey of the period 1966-1988. Recent textbooks are Khuri and Cornell (1996) and Myers and Montgomery (2002). Recent software is Stat-Ease's 'Design-Ease' and 'Design-Expert'.

This RSM has also been applied to *simulation*; see Bartz-Beielstein (2006), Irizarry et al. (2001), Kleijnen (1998), Rosen et al. (2006), Yang and Tseng (2002). A case study of RSM for deterministic simulation is Ben-Gal and Bukchin (2002). RSM is also discussed in the classic discrete-event simulation textbook by Law and Kelton (2006, pp. 646-655).

As I have already stated in Section 1, the simplest optimization problem concerns the minimization of the expected value of a single (univariate) simulation output :

$$\min_{\mathbf{z}} E(w_0|\mathbf{z}, \mathbf{r}_0) \tag{1}$$

where

$E(w_0|\mathbf{z}, \mathbf{r}_0)$ is the goal (or objective) output of the (random; see $\mathbf{r}_0$) simulation model, which is to be minimized through the choice of $\mathbf{z}$;

$\mathbf{z} = (z_1, \ldots, z_k)'$ where z_j $(j = 1, \ldots k)$ denotes the j^{th} original (non-coded, non-standardized) input of the simulation program;

$\mathbf{r}_0$ is the vector with the seeds of the PRN streams (this vector may consist of a single element).

Classic RSM (applied to real-world or simulated systems) has the following *characteristic*s.

- RSM is an *optimization heuristic* that tries to estimate the input combination that minimizes a given goal function; see (1) above. Because RSM is a heuristic, success is not guaranteed (see below).
- RSM is a *stepwise* (multi-stage) method.
- In these steps, RSM uses first-order and second-order *polynomial* regression (meta)models (response surfaces). RSM assumes that the responses have *white noise*; i.e., the residuals e are Normally, Independently, and Identically Distributed (NIID) with zero mean and constant variance: $e \sim NIID(0, \sigma^2)$. Ordinary Least Squares (OLS) then gives the Best Linear Unbiased Estimator (BLUE); see Kleijnen (2006).
- To fit (estimate, calibrate) these first-order polynomials, RSM uses *classic designs* of resolution III; for second-order polynomials, RSM uses a Central Composite Design (CCD); details on these designs are given in Myers and Montgomery (2002).
- To determine in which direction the inputs will be changed in a next step, RSM uses the *gradient* that is implied by the first-order polynomial fitted in the current step. This gradient is used in the mathematical (not statistical) technique of *steepest descent* (or steepest ascent, in case the output is to be maximized, not minimized; details follow below).
- In the final step, RSM applies the mathematical technique of *canonical analysis* to the second-order polynomial metamodel, to examine the shape of the optimal (sub)region: does that region have a unique minimum, a saddle point, or a ridge (stationary points)?

More specifically, classic RSM consists of the following eight *steps* (also see Figure 1 below, which gives an example with a single goal function and two constrained random outputs; these constraints vanish in classic RSM).

1. The analysts begin by selecting a *starting point*. They may select the input combination currently used in practice if the simulated system already exists. Otherwise, they should use intuition and prior knowledge (as in many other metaheuristics).
2. For the *neighborhood* of this starting point, the analysts explore the Input/Output (I/O) behavior of the simulated system. This behavior is approximated through a local first-order polynomial (as the Taylor series expansion suggests). Hence the analysts need to estimate the intercept β_0 and the k main (first-order) effects β_j with $j = 1, \ldots, k$. Therefore they use a resolution-III design. Unfortunately, there are no general guidelines to determine the appropriate *size* of this local area; again, intuition and prior knowledge are important.
3. To decide on the next subarea to be explored by simulation, the analysts follow the *steepest descent path*, which uses the local gradient. For

example, if the estimated first-order effects are such that $\widehat{\beta}_1 >> \widehat{\beta}_2$, then they obviously decrease z_1 much more than z_2. Unfortunately, the steepest descent method is *scale dependent*; i.e., linear transformations of the inputs affect the search direction (see Myers and Montgomery (2002, pp. 218-220)). Fortunately, Kleijnen et al. (2004, 2006) developed a scale-independent variant, which I shall present at the end of this section.

4. Unfortunately, the steepest descent technique does not quantify the *step size* along its path. The analysts may therefore try some value for the step size. If that value yields an inferior simulation output (higher instead of lower output), then they may reduce the step size. Otherwise, they take one more step in the current steepest descent direction. (There are more sophisticated mathematical procedures for selecting step sizes; see Safizadeh and Signorile (1994), and the following section, Section 3.)
5. After a number of steps along the steepest descent path, the simulation output will *deteriorate* (increase instead of decrease), because the first-order polynomial is only a local approximation of the implicit I/O function defined by the simulation model itself. When this deterioration happens, the analysts simulate the $n > k$ factor combinations specified by a *resolution-III design* centered around the best point found so far. (The analysts may use the same coded design as in step 2, but translate that design into different values for the original variables; the best combination found so far, may be one of the corner points of the design; see Figure 1 below.) Next the analysts estimate the first-order effects in the new local polynomial approximation And so their search continues.
6. However, it is intuitively clear that a *plane* (implied by the most recent local first-order polynomial) cannot adequately represent a *hill top* (when searching for the maximum; the analogue holds for the minimum). So in the neighborhood of the optimum, a first-order polynomial shows serious *lack of fit*. A popular and simple diagnostic measure is the coefficient of determination R^2. A related diagnostic tests whether all estimated first-order effects (and hence the gradient) are zero. Instead of these diagnostics, the analysts might use cross-validation. If the most recently fitted first-order polynomial turns out to be inadequate, then the analysts fit a *second-order polynomial*. To estimate this metamodel, they simulate the combinations specified by a CCD.
7. From this second-order polynomial, the analysts estimate the optimal values of the decision variables by straightforward *differentiation* or by more sophisticated *canonical analysis*; see Myers and Montgomery (2002, p. 208).
8. If time permits, then the analysts may try to escape from a local minimum and *restart* their search from a different initial local area—which brings them back to Step 1.

In step 3, I mentioned a variant of steepest descent. This so-called *Adapted Steepest Descent* (ASD) accounts for the covariances between the

elements of the estimated gradient $\widehat{\boldsymbol{\beta}}_{-0} = (\widehat{\beta}_1, \ldots, \widehat{\beta}_k)'$, where the subscript -0 means that the intercept $\widehat{\beta}_0$ of the estimated first-order polynomial vanishes in the estimated gradient so $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_0, \widehat{\boldsymbol{\beta}}_{-0})'$.

This $\mathbf{cov}(\widehat{\boldsymbol{\beta}}_{-0})$ follows from the (classic) white noise assumption:

$$\mathbf{cov}(\widehat{\boldsymbol{\beta}}) = \sigma_w^2(\mathbf{Z}'\mathbf{Z})^{-1} = \sigma_w^2 \begin{pmatrix} a & \mathbf{b}' \\ \mathbf{b} & \mathbf{C} \end{pmatrix} \tag{2}$$

where

σ_w^2 denotes the variance of the (goal) simulation output w;

$\mathbf{Z}$ the $N \times (1+k)$ matrix of explanatory regression variables including the column with N one's;

$N = \sum_{i=1}^{n} m_i$ the total number of simulation runs;

n the number of simulated input combinations;

m_i the number of Identically and Independently Distributed (IID) replicates for combination i;

a a scalar;

$\mathbf{b}$ a k-dimensional vector;

$\mathbf{C}$ a $k \times k$ matrix such that $\mathbf{cov}(\widehat{\boldsymbol{\beta}}_{-0}) = \sigma_w^2\mathbf{C}$.

Note that $\mathbf{Z}$'s first column corresponds with the intercept β_0. Furthermore, $\mathbf{Z}$ is determined by a resolution-III design (so $n > k$), transformed into the original values of the inputs in the local area. To save computer time, only the center of the local area may be replicated (the center is not part of the resolution-III design). Replicates use the same input combination $\mathbf{z}_i (i = 1, \ldots n)$.

The variance σ_w^2 in (2) is estimated through the Mean Squared Residuals (MSR):

$$\widehat{\sigma}_w^2 = \frac{\sum_{i=1}^{n} \sum_{r=1}^{m_i} (w_{i;r} - \widehat{y}_i)^2}{\sum_{i=1}^{n} m_i - (k+1)} \tag{3}$$

where $\widehat{y} = \mathbf{z}_i'\widehat{\boldsymbol{\beta}}$; also see Myers and Montgomery (2002).

It is easy to prove that the predictor variance $var(\widehat{y}|\mathbf{z})$ increases as $\mathbf{z}$ (point to be predicted) moves further away from the local area where the gradient is estimated. The point with the minimum predictor variance is the point $-\mathbf{C}^{-1}\mathbf{b}$ (with $\mathbf{C}$ and $\mathbf{b}$ defined below equation 2).

The new point to be simulated is

$$\mathbf{d} = -\mathbf{C}^{-1}\mathbf{b} - \lambda\mathbf{C}^{-1}\widehat{\boldsymbol{\beta}}_{-0} \tag{4}$$

where

$-\mathbf{C}^{-1}\mathbf{b}$ is the point where the local search starts, namely the point with the minimum variance locally;

λ is the step size;

$\mathbf{C}^{-1}\widehat{\boldsymbol{\beta}}_{-0}$ is the (classic) steepest descent direction $\widehat{\boldsymbol{\beta}}_{-0}$—adapted for $\mathbf{cov}(\widehat{\boldsymbol{\beta}}_{-0})$.

Accounting for $\mathbf{cov}(\widehat{\boldsymbol{\beta}}_{-0})$ gives a scale independent search direction, which is an important characteristic for both practitioners and researchers.

This search direction in general performs better than steepest descent; see Kleijnen et al. (2004, 2006).

3 GRSM: multiple outputs and constraints

In practice, simulation models have *multiple* responses (multivariate output). Several approaches to solve the resulting issues are surveyed by Rosen et al. (2006). Furthermore, the RSM literature also offers a few approaches for such situations; see the surveys in Angün et al. (2006) and Khuri (1996). However, I find these approaches less attractive than the following approach, called GRSM.

Assume that one simulation output should be minimized, while all the other outputs must satisfy given constraints. More specifically, GRSM has the following characteristics.

- GRSM generalizes the steepest descent search direction (applied in classic RSM), using the 'affine scaling search direction' and borrowing ideas from *interior point* methods (a variation on Karmarkar's algorithm) in mathematical programming; see Barnes (1986). This novel search direction moves faster to the optimum than steepest descent, since the GRSM search avoids creeping along the boundary of the feasible area (this feasible area is determined by the constraints on the random outputs and the deterministic inputs; see below.) Moreover, this search direction is scale independent.
- GRSM uses its search direction *iteratively* (as classic RSM does). Because this heuristic is developed for expensive simulation experiments, the search should quickly reach a neighborhood of the true optimum.
- Though GRSM has been developed for *random* simulations, it can easily be adapted for *deterministic* simulations and *real-world* (non-simulated) systems (analogous to classic RSM).

Formally, GRSM extends the classic RSM problem formulated in (1) to the following *constrained nonlinear random optimization problem*:

$$\min_{\mathbf{z}} E(w_0|\mathbf{z}, \mathbf{r}_0) \tag{5}$$

such that the other $(r-1)$ random outputs satisfy the constraints

$$E(w_{h'}|\mathbf{z}, \mathbf{r}_0) \geq a_{h'} \text{ with } h' = 1, \ldots, r-1 \tag{6}$$

and the k deterministic inputs satisfy the *box constraints*

$$l_j \leq z_j \leq u_j \text{ with } j = 1, \ldots, k. \tag{7}$$

An example is the following inventory simulation. The sum of the expected inventory carrying costs and ordering costs should be minimized, The expected service percentage (or fill rate) should be at least (say) 90%

so $a_1 = 0.9$ in (6). Both the reorder quantity $z_1(= Q)$ and the reorder level $z_2(= s)$ should be non-negative, so $z_1 \geq 0$ and $z_2 \geq 0$. (Note the similarity of the constraints on the random outputs and the deterministic inputs.) Stricter input constraints may be formulated; for example, the reorder level should at least cover the expected demand during the expected order lead time. Input constraints more complicated than these box constraints (namely, geometry constraints) are discussed in Stinstra et al. (2003).

Analogously to the first steps of classic RSM, GRSM locally approximates the multivariate I/O function by r univariate first-order polynomials:

$$\mathbf{y}_h = \mathbf{Z}\boldsymbol{\beta}_h + \mathbf{e}_h \text{ with } h = 0, \ldots r-1. \tag{8}$$

Like RSM, GRSM assumes that *locally* the white noise assumption holds. The following OLS estimators are then the BLUE:

$$\hat{\boldsymbol{\beta}}_h = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{w}_h \text{ with } h = 0, \ldots, r-1 \tag{9}$$

Then $\hat{\boldsymbol{\beta}}_0$ (OLS estimator for first-order polynomial approximation of goal function) and the goal function (5) result in

$$\min_{\mathbf{z}} \widehat{\boldsymbol{\beta}}_{0;-0}\mathbf{z} \tag{10}$$

where $\widehat{\boldsymbol{\beta}}_{0;-0}$ denotes the OLS estimate of the local regression parameters for the goal output (which explains the first subscript 0) excluding the intercept (which explains the second subscript -0); in other words, $\widehat{\boldsymbol{\beta}}_{0;-0} = (\widehat{\beta}_{0;1}, \ldots, \widehat{\beta}_{0,k})'$ is the *estimated local gradient* of the goal function.

The $(r-1)$ estimates $\hat{\boldsymbol{\beta}}_{h'}$ in (9) combined with the original output constraints (6) give

$$\widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{z} \geq c_{h'} \text{ with } h' = 1, \ldots, r-1 \tag{11}$$

where $\widehat{\boldsymbol{\beta}}_{h';-0} = (\widehat{\beta}_{h';1}, \ldots, \widehat{\beta}_{h',k})'$ denotes the estimated local gradient of constraint function h', and $c_{h'} = a_{h'} - \widehat{\beta}_{h';0}$ denotes the modified right-hand side of this constraint function. The box constraints in (7) remain unchanged.

Now the $(r-1)$ k-dimensional vectors $\widehat{\boldsymbol{\beta}}_{h';-0}$ in (11) are collected in the $(r-1) \times k$ matrix called $\mathbf{B}$. Likewise, the $(r-1)$ elements $c_{h'}$ are collected in the vector $\mathbf{c}$. And the k-dimensional vectors with the non-negative *slack variables* $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{v}$ are introduced. Altogether this gives

$$\begin{array}{ll} \text{minimize} & \widehat{\boldsymbol{\beta}}'_{0;-0}\mathbf{z} \\ \text{subject to} & \mathbf{Bz} - \mathbf{s} = \mathbf{c} \\ & \mathbf{z} + \mathbf{r} = \mathbf{u} \\ & \mathbf{z} - \mathbf{v} = \mathbf{l}. \end{array} \tag{12}$$

This (local) optimization problem is *linear* in the decision variables $\mathbf{z}$. GRSM does not solve this LP problem, but uses this problem only to derive the following novel *search direction* $\mathbf{d}$:

$$\mathbf{d} = -\left(\mathbf{B}'\mathbf{S}^{-2}\mathbf{B} + \mathbf{R}^{-2} + \mathbf{V}^{-2}\right)^{-1} \widehat{\boldsymbol{\beta}}_{0;-0} \tag{13}$$

where $\mathbf{S}$, $\mathbf{R}$, and $\mathbf{V}$ are diagonal matrixes with as main-diagonal elements the current estimated slack vectors $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{v}$ in (12); the last factor ($\widehat{\boldsymbol{\beta}}_{0;-0}$) is the estimated classic steepest ascent direction. GRSM's search direction can be proven to be scale independent.

Note that as the value of a slack variable in (13) decreases (so the corresponding constraint gets tighter), the GRSM search direction deviates more from the steepest descent direction. Possible singularity of the various matrices in (13) is discussed in Angün et al. (2006).

Following the search direction (or path) defined by (13), GRSM must decide on the *step size* (say) λ along this path. An explicit step size assuming that the local metamodel (11) holds *globally* is

$$\lambda = 0.8 \min_{h'} \left[\frac{c_{h'} - \widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{z}_c}{\widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{d}} \right] \tag{14}$$

where the factor 0.8 is chosen to decrease the probability that the local metamodel is misleading when applied globally; $\mathbf{z}_c$ denotes the current input combination, so the new combination becomes $\mathbf{z}_c + \lambda\mathbf{d}$. Obviously, the box constraints (7) for the deterministic inputs hold globally, so it is easy to check the solution in (14) against these constraints.

Analogously to classic RSM, GRSM proceeds *stepwise*; i.e., after each step along the search path, the following hypotheses are tested:

1. $w_0(\mathbf{z}_c + \lambda\mathbf{d})$ (simulation output of new combination) is *no improvement* over $w_0(\mathbf{z}_c)$ (output of old combination); i.e. this step increases the goal output w_0 (pessimistic null-hypothesis):

$$H_0 : E[w_0(\mathbf{z}_c + \lambda\mathbf{d})] \geq E[w_0(\mathbf{z}_c)]. \tag{15}$$

2. This step is *feasible*; i.e., the new solution satisfies the $(r-1)$ constraints in (6):

$$H_0 : E(w_{h'}|\mathbf{z}, \mathbf{r}_0) \geq a_{h'} \text{with } h' = 1, \ldots, r-1. \tag{16}$$

To test these hypotheses, I propose the following statistical procedures (more complicated parametric bootstrapping is used by Angün et al. 2006, which permits non-normality and tests the relative improvement $w_0(\mathbf{z}_c + \lambda\mathbf{d})/w_0(\mathbf{z}_c)$ and the relative slacks $s_{h'}(\mathbf{z}_c + \lambda\mathbf{d})/s_{h'}(\mathbf{z}_c)$).

Sub 1: To test (15), the classic Student t statistic may be applied. A paired t statistic may be applied if Common Random Numbers (CRN) are used to obtain the two simulation outputs $w_0(\mathbf{z}_c + \lambda\mathbf{d})$ and $w_0(\mathbf{z}_c)$. To estimate the standard error of their difference, $m \geq 2$ replicates suffice:

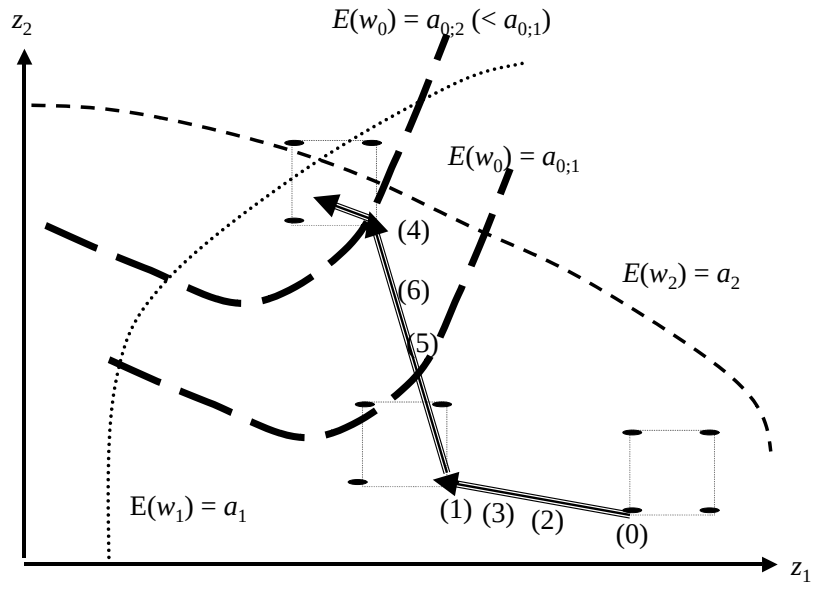

Fig. 1 GRSM example

the t statistic has $m-1$ degrees of freedom. The hypothesis is rejected if significant improvement is observed.

Sub 2: Again a t statistic with $m-1$ degrees of freedom may be applied. Because $r-1$ hypotheses are implied by (16), Bonferroni's inequality may be used (i.e., divide the classic α value by the number of tests). Recent references on this inequality are given in Gordon (2006).

Actually, a better solution may lie in between $\mathbf{z}_c$ (old combination) and $\mathbf{z}_c + \lambda \mathbf{d}$ (new combination at 'maximum' step size). Therefore GRSM uses *binary search*; i.e., it simulates a combination that lies halfway these two combinations (and is still on the search path). Actually, this halving of the stepsize may be applied a number of times—accounting for the limited computer budget.

Next GRSM proceeds analogously to classic RSM. So around the best combination found so far, it selects a new local area. Again a resolution-III design selects new simulation runs. Again, only the new center may be replicated $m > 1$ times. And r first-order polynomials are fitted, which gives a *new* search direction. And so on.

Angün et al. (2006) applied GRSM to two examples, namely Bashyam and Fu (1998)'s inventory simulation with a service constraint so the solution is unknown, and an artificial example with known solution. The results of these examples are encouraging, as GRSM found solutions that were both feasible and gave drastically lower goal functions. Figure 1 gives an example, which deserves the following comments.

There are two inputs; see the two axes labeled z_1 and z_2 in the figure. There is one goal function; the figure shows only two contour plots (iso-costs functions), namely $E(w_0) = a_{0;1}$ and $E(w_0) = a_{0;2}$ with $a_{0;2} < a_{0;1}$. There are two constrained random outputs; see $E(w_1) = a_1$ and $E(w_2) = a_2$. The search starts in the lower right local area, where a 2^2 design is executed; see the four elongated points. This design and the (not shown) replicates give a search direction; see the arrow leaving from point (0). The maximum step size along this path takes the search from point (0) to point (1). The binary search takes the search back to point (2), and next to point (3). Because the best point so far turns out to be point (1), the 2^2 design is simulated at the local area with this point as one of its points. This design gives a new search direction, which avoids the boundary. The maximum step size now takes the search to point (4). The binary search takes the search back to point (5), and next to point (6). Because the best point so far is now point (4), the 2^2 design is simulated at the local area with this point as one of its points. A new search direction is estimated; etc. (the remaining search is not displayed).

4 Testing an estimated optimum: Karush-Kuhn-Tucker conditions

By definition, it is uncertain whether the optimum estimated by a (meta)heuristic (such as GRSM) is close enough to the true optimum. In *deterministic* nonlinear mathematical programming, the *Karush-Kuhn-Tucker* (KKT) first-order optimality conditions have been derived; see, for example, Gill et al. (2000).

Figure 2 illustrates the same type of problem as the one in Figure 1. In the present example there is again a goal function $E(w_0)$, for which three contour plots are displayed corresponding to the values 66, 76, and 96; also see (5). There are two constrained simulation outputs, namely $E(w_1) = 4$ and $E(w_2) = 9$; also see (6). The optimum combination is point A. Points B and C lie on the boundary $E(w_2) = 9$; point D lies on the boundary $E(w_1) = 4$. Obviously, point D is far away from the optimum combination A. The figure also displays the (local) gradients at these four points for the goal function and the *binding constraint*; i.e., the constraint with a zero slack value in (6). These gradients are perpendicular to the local tangent lines; those lines are shown only for the binding constraint—not for the goal function. At the optimum the gradients for the goal function and the binding constraint coincide; at point D these two gradients point in very different directions.

The characteristic illustrated in this figure is formalized by the KKT conditions:

$$\boldsymbol{\beta}_{0;-0} = \mathbf{B}_{J;-0}\boldsymbol{\lambda} \tag{17}$$

where

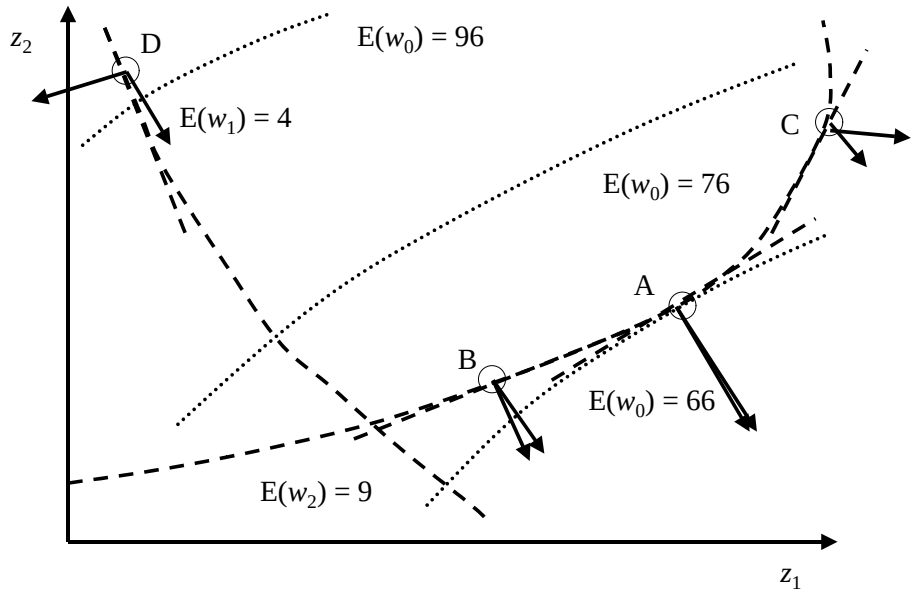

Fig. 2 A constrained nonlinear random optimization problem

$\boldsymbol{\beta}_{0;-0}$ denotes the k-dimensional vector with the (deterministic) gradient of the goal function (also see equation 10);

$\mathbf{B}_{J;-0}$ denotes the $k \times J$ matrix with the gradients $\boldsymbol{\beta}_{h;-0}$ of the J binding constraints (if the constrained simulation outputs are relabeled such that the first binding constraint has label 1, the second binding constraint has label 2, and so on, then $h = 1, \ldots, J$; at the four points labeled A through D in Figure 2 the matrix $\mathbf{B}_J$ has only one column; this column consists of the components of the gradient of the constraint that is binding at the specific point);

$\boldsymbol{\lambda}$ denotes the J-dimensional vector with the non-negative Lagrange multipliers.

Figure 2 shows that point A satisfies (17); point B has two gradients that point in different but similar directions—and so does point C. Point D has two gradients that point in completely different directions.

Note: If the optimum occurs *inside* the feasible area, then there are no binding constraints. The KKT conditions then reduce to the condition that the goal gradient is zero. Classic RSM tests for a zero gradient, estimated from a second-order polynomial; see Step 7 in Section 2. This test may use a classic F-test (see Section 2 and Myers and Montgomery (2002)). In this article, I do not consider such situations any further.

Unfortunately, in *random* simulation the gradients must be *estimated.* Moreover, to check which constraints are binding, the slacks of the constraints must be estimated. This estimation turns the KKT conditions (17) into a problem of nonlinear statistics.

Angün and Kleijnen (2006) derived an asymptotic test to check whether the optimum has indeed been found. Bettonvil et al. (2006) derived an alternative, bootstrap test. I focus on the latter test, because it is simpler (the former test uses the so-called Delta method and a generalized form of the so-called Wald statistic) and it allows a small number of replicates (as is the case in expensive simulation). Both tests assume a problem like the one formulated in the preceding section; i.e., there is one random simulation output to be minimized and there are $r-1$ constrained random simulation outputs; see (5) and (6).

Both tests assume that *estimated gradients* are available. RSM and GRSM do give estimated gradients; most metaheuristics do not estimate gradients. However, when the latter metaheuristics are applied, then at the presumed end of the search a local experiment may be performed to estimate the gradients and use these gradients as a stopping criterion—instead of using rather arbitrary criteria such as a prefixed computer budget!

Note: Whenever a metaheuristic is used to estimate gradients while treating the simulation model as a *black box*, the analysts should not change one factor at a time followed by some type of finite differencing (such an approach is proposed in, for example, Spall (2003)). Instead, the analysts should use classic designs to fit first-order or second-order polynomials locally; for example, the tangent lines in Figure 2 may be interpreted as first-order polynomials. To fit such polynomials, classic RSM uses highly efficient resolution-III designs and a CCD (see Section 2 and also Joshi et al. (1998)). Obviously, the estimated gradient is biased if second-order effects are important and yet a first-order polynomial is fitted.

As in classic RSM, let us assume *locally constant (co)variances* for each of the r simulation outputs; i.e., when moving to a new local area, the (co)variances may change. For example, the points A through D in Figure 4 do not have the same variance for the goal output. Under these assumptions, OLS applied per univariate simulation output gives the BLUE, $\hat{\boldsymbol{\beta}}_h$ ($h = 0, 1, \ldots, r-1$) defined in (9). These OLS estimators then have the following estimated covariance matrix:

$$\widehat{\mathbf{cov}}(\hat{\boldsymbol{\beta}}_h, \hat{\boldsymbol{\beta}}_{h'}) = \widehat{\mathbf{cov}}(w_h, w_{h'}) \otimes (\mathbf{Z}'\mathbf{Z})^{-1} \ (h, h' = 0, \ldots, r-1) \qquad (18)$$

where $\otimes$ is the well-known 'Kronecker product' operator and $\widehat{\mathbf{cov}}(w_h, w_{h'})$ is an $r \times r$ matrix with the classic estimators of the (co)variances based on the m replicates at the local center:

$$\widehat{\mathbf{cov}}(w_h, w_{h'}) = (\widehat{\sigma}_{h;h'}) = \left(\sum_{l=1}^{m} (w_{h;l} - \overline{w}_h)(w_{h';l} - \overline{w}_{h'})\right) \frac{1}{m-1}. \qquad (19)$$

The Kronecker product implies that $\widehat{\mathbf{cov}}(\hat{\boldsymbol{\beta}}_h, \hat{\boldsymbol{\beta}}_{h'})$ is an $rq \times rq$ matrix with q denoting the number of regression parameters (e.g., $q = 1+k$ in a first-order polynomial), formed from the $r \times r$ matrix $\widehat{\mathbf{cov}}(w_h, w_{h'})$ by multiplying each of its elements by the entire $q \times q$ matrix $(\mathbf{Z}'\mathbf{Z})^{-1}$ (in equation 2, $\mathbf{Z}$ was

an $N \times (1+k)$ matrix). The matrix $\widehat{\mathbf{cov}}(w_h, w_{h'})$ is singular if $m \leq r$; for example, the case study in Kleijnen (1993) has $r = 2$ response types and $k = 14$ inputs so $m \geq 3$ replicates of the center point are required.

Another reason for replicating the center point is that this point is used to test whether a constraint is binding in the current local area (see equation 20 below). The center point is more representative of the local behavior than any of the other points of the design. Classic RSM also uses replication of the center point when using a CCD (for estimating a second-order polynomial).

Let us further assume that the r-variate simulation output is *multivariate Gaussian*. Then (as in classic RSM) the *validity* of the local metamodel may be tested through the classic lack-of–fit F statistic; see Myers and Montgomery (2002). This test also assumes that no CRN are applied. In GRSM there are multiple simulation responses, so this classic test is combined with *Bonferroni's inequality*; i.e., the classic type-I error rate α is replaced by the 'experimentwise' or 'familywise' error rate α/r.

If the metamodel is rejected, then there are two options: (i) Decrease the local area; for example, halve each factor's range. (ii) Increase the order of the polynomial; for example, switch from a first-order to a second-order polynomial. I do not explore these options further in this article.

Testing the KKT conditions in random simulation implies testing the following three null-hypotheses, denoted by the superscripts (1) through (3):

1. The current solution is feasible and at least one constraint is binding; see (6):

$$H_0^{(1)} : E(w_{h'} | \mathbf{x} = \mathbf{0}, \mathbf{r}_0) = a_{h'} \text{ with } h' = 1, \ldots, r-1 \tag{20}$$

   where $\mathbf{x} = \mathbf{0}$ corresponds with the center point expressed in the coded (standardized) inputs.
2. The expected value of the estimated local gradient equals the expected value of the product of the estimated gradients of the simulation outputs in the binding constraints and the Lagrange multipliers; see (17):

$$H_0^{(2)} : E(\widehat{\boldsymbol{\beta}}_{0;-0}) = E(\widehat{\mathbf{B}}_{J;-0}\widehat{\boldsymbol{\lambda}}). \tag{21}$$

3. The Lagrange multipliers in (21) are non-negative:

$$H_0^{(3)} : E(\widehat{\boldsymbol{\lambda}}) \geq \mathbf{0}. \tag{22}$$

Each of these three hypotheses requires multiple tests, so Bonferroni's inequality is applied. Moreover, these three hypotheses are tested sequentially, so it is hard to control the final type-I and type-II error probabilities. However, classic RSM has the same type of problems, and nevertheless, it has acquired a track record in practice.

Sub 1: To test the hypothesis (20), the classic Student t statistic may be used:

$$t_{m-1}^{(h')} = \frac{w_{h'}(\mathbf{x} = \mathbf{0}, \mathbf{r}_0) - a_{h'}}{\sqrt{\widehat{\sigma}_{h';h'}/m}} \text{with } h' = 1, \ldots, r-1 \tag{23}$$

where both the numerator and the denominator are based on the m replicates at the local center point; see (19).

Note: To save simulation runs, a local experiment should start at its center point including replicates. If it turns out that either no constraint is binding or at least one constraint is violated, then the other hypotheses need not be tested so the remainder of the local design is not simulated.

The t statistic in (23) may give the following three different results:

- The statistic is *significantly positive*; i.e., the constraint for output h' is not binding. If none of the $(r-1)$ constraints is binding, then the optimal solution is not yet found (assuming that at the optimum at least one constraint is binding; otherwise, classic RSM applies). The search for better solutions continues (also see Section 3).
- The statistic is *significantly negative*; i.e., the current local area does not give feasible solutions so the optimal solution is not yet found. The search should back up into the feasible area.
- The statistic is *non-significant*; i.e., the current local area gives feasible solutions, and the constraint for output h' is binding. The gradient of this response is then included in $\widehat{\mathbf{B}}_J$; see (21). And the KKT test proceeds as follows.

Sub 2: The hypothesis (21) states that the goal gradient is a *linear* combination of the gradients of the binding constraints. Such a combination may be estimated through *OLS*, using $\widehat{\mathbf{B}}_{J;-0}$ as the matrix of explanatory variables:

$$\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0} = \widehat{\mathbf{B}}_{J;-0}(\widehat{\mathbf{B}}'_{J;-0}\widehat{\mathbf{B}}_{J;-0})^{-1}\widehat{\mathbf{B}}'_{J;-0}\widehat{\boldsymbol{\beta}}_{0;-0} = \widehat{\mathbf{B}}_{J;-0}\widehat{\boldsymbol{\lambda}} \tag{24}$$

with $\widehat{\boldsymbol{\lambda}} = (\widehat{\mathbf{B}}'_{J;-0}\widehat{\mathbf{B}}_{J;-0})^{-1}\widehat{\mathbf{B}}'_{J;-0}\widehat{\boldsymbol{\beta}}_{0;-0}$.

The *validity* of this linear approximation may be quantified through the k-dimensional vector of residuals

$$\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0}) = \widehat{\widehat{\boldsymbol{\beta}}}_{0;-0} - \widehat{\boldsymbol{\beta}}_{0;-0}. \tag{25}$$

The hypothesis (21) implies $E(\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0})) = \mathbf{0}$. This hypothesis involves the product of multivariates, so standard tests do not apply. Therefore *bootstrapping* may be used. Distribution-free bootstrapping does not apply because in expensive simulation only the center point is replicated a few times. Instead, *parametric bootstrapping* should be used; i.e., a specific distribution type is assumed and its parameters are estimated from the simulation's I/O data at hand (so the bootstrap is called 'data driven'; it is also known as the 'Monte Carlo' method). Like in classic RSM, the type of distribution assumed for testing the KKT conditions is the Gaussian distribution.

Figure 3 shows the following three layers of models:

1. The simulation model, which is treated as a black box. Like in Figure 2, only two simulation inputs and three simulation outputs are assumed.

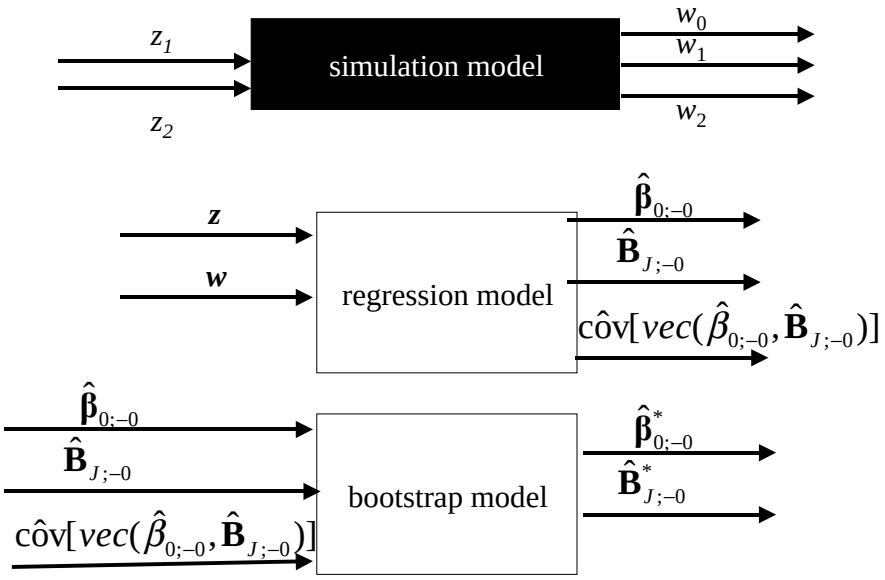

Fig. 3 I/O of simulation, regression, and bootstrap models

2. The regression metamodel, which uses the simulation I/O data $(\mathbf{Z}, \mathbf{w})$ as input and estimates the gradients of the goal response ($\widehat{\boldsymbol{\beta}}_{0;-0}$) and of the constrained responses including the binding constraints collected in $\widehat{\mathbf{B}}_{J;-0}$. The regression analysis also estimates $\widehat{\mathbf{cov}}(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0})$ (covariance matrix of these estimated gradients).
3. The bootstrap model, which uses the regression output ($\widehat{\boldsymbol{\beta}}_{0;-0}$, $\widehat{\mathbf{B}}_{J;-0}$, $\widehat{\mathbf{cov}}(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0})$) as parameters of the multivariate normal distribution of its output $\widehat{\boldsymbol{\beta}}^*_{0;-0}$ and $\widehat{\mathbf{B}}^*_{J;-0}$ where the superscript $*$ denotes bootstrapped values.

More specifically, the bootstrap procedure consists of the following four steps:

1. Use the *Monte Carlo* method to sample $vec(\widehat{\boldsymbol{\beta}}^*_{0;-0}, \widehat{\mathbf{B}}^*_{J;-0})$, which is a $(k + kJ)$-dimensional vector formed by 'stapling' (stacking) the k-dimensional goal gradient vector and the J k-dimensional vectors of the $k \times J$ matrix $\widehat{\mathbf{B}}^*_{J;-0}$:

$$vec(\widehat{\boldsymbol{\beta}}^*_{0;-0}, \widehat{\mathbf{B}}^*_{J;-0}) \sim N(vec(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0}), \widehat{\mathbf{cov}}(vec(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0}))) \tag{26}$$

   where $\widehat{\mathbf{cov}}(vec(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0}))$ is the $(k+kJ)\times(k+kJ)$ matrix computed through (18).
2. Use the bootstrap values resulting from Step 1, to compute the *OLS* estimate of the bootstrapped goal gradient using the bootstrapped gra-

dients of the binding constraints as explanatory variables; i.e., use (24) adding the superscript $*$ to all random variables resulting in $\widehat{\widehat{\boldsymbol{\beta}}}^*_{0;-0}$ and $\widehat{\boldsymbol{\lambda}}^*$.

3. Use $\widehat{\widehat{\boldsymbol{\beta}}}^*_{0;-0}$ from Step 2 and $\widehat{\boldsymbol{\beta}}^*_{0;-0}$ from Step 1 to compute the *bootstrap residual* $\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}^*_{0;-0}) = \widehat{\widehat{\boldsymbol{\beta}}}^*_{0;-0} - \widehat{\boldsymbol{\beta}}^*_{0;-0}$ (analogous to equation 25). Furthermore, determine whether any of the *bootstrapped Lagrange multipliers* $\widehat{\boldsymbol{\lambda}}^*$ (found in Step 2) is negative; i.e., augment a counter (say) c^* with the value 1 if this event occurs.
4. *Repeat* the preceding three steps (say) 1000 times (this is known as the 'bootstrap sample size'). This gives the Estimated Density Function (EDF) of the bootstrapped residuals per input j ($j = 1, \ldots, k$) $\widehat{\mathbf{e}}(\widehat{\widehat{\beta}}^*_{0;-0;j})$ and the final value of the counter c^*. Reject the hypothesis in (21) if this EDF implies a two-sided $(1 - \alpha/(2k))$ confidence interval that does not cover the value 0 (the factor k is explained by Bonferroni's inequality). Reject the hypothesis in (22) if the fraction $c^*/1000$ is significantly higher than 50% (if the true Lagrange multiplier is only 'slightly' larger than zero, then 'nearly' 50% of the bootstrapped values is negative). To test the latter fraction, the binomial distribution may be approximated though the normal distribution with mean 0.50 and variance $(0.50 \times 0.50)/1000 = 0.00025$.

The numerical examples in Bettonvil et al. (2006) are encouraging:

1. The classic t test for zero slacks and the classic F test for lack of fit perform as expected.
2. The new bootstrap tests give observed type I error rates close to the prespecified (nominal) rates; the type II error rate (complement of the power) decreases as the input combination tested moves farther away from the true optimum (see the points A through D in Figure 2).

5 Conclusions

In this article, I first summarized classic RSM, assuming a single response variable. I added the Adapted Steepest Ascent (ASA) search direction, which improves the classic Steepest Ascent.

Next, I summarized GRSM for simulation with multivariate responses, assuming that one response is to be minimized while all the other responses must meet given constraints. Moreover, the (deterministic) inputs must satisfy given box constraints.

Finally, I summarized a procedure for testing whether an estimated optimum is truly optimal—using the KKT conditions. This procedure combines classic tests and bootstrapped tests.

Further research is needed to investigate the performance of GRSM relative to other metaheuristics.

References

1. Angün, E., D. den Hertog, G. Gürkan, and J.P.C. Kleijnen (2006), Response surface methodology with stochastic constraints for expensive simulation. Working Paper, Tilburg University, Tilburg, Netherlands
2. Angün, E. and J.P.C. Kleijnen (2006), An asymptotic test of optimality conditions in multiresponse simulation-based optimization. Working Paper, Tilburg University, Tilburg, Netherlands
3. Barnes, E. R. (1986), A variation on Karmarkar's algorithm for solving linear programming problems. *Mathematical Programming*, 36, pp. 174-182
4. Bartz-Beielstein, T. (2006), *Experimental research in evolutionary computation; the new experimentalism*. Springer, Berlin
5. Bashyam, S. and M. C. Fu (1998), Optimization of (s, S) inventory systems with random lead times and a service level constraint. *Management Science*, 44, pp. 243-256
6. Ben-Gal, I. and J. Bukchin (2002), Ergonomic design of working environment via rapid prototyping tools and design of experiments, *IIE Transactions*, 34, no. 4, pp. 375-391
7. Bettonvil, B., E. del Castillo, and J.P.C. Kleijnen (2006), Statistical testing of optimality conditions in multiresponse simulation-based optimization. Working Paper, Tilburg University, Tilburg, Netherlands
8. Box, G.E.P. and K.B. Wilson (1951), On the experimental attainment of optimum conditions. *Journal Royal Statistical Society, Series B*, 13, no. 1, pp. 1-38
9. Downing, D.J. , R.H. Gardner and F.O. Hoffman (1985), An examination of Response-Surface Methodologies for uncertainty analysis in assessment models. *Technometrics*, 27, no. 2, pp. 151-163 (Discussion: 1986, 28, no. 1, pp. 91-93)
10. Gill, P. E., W. Murray, and M. H. Wright (2000). *Practical optimization, 12th edition*. Academic Press, London
11. Gordon, A.Y. (2006), Unimprovability of the Bonferroni procedure in the class of general step-up multiple testing procedures. *Statistics & Probability Letters,* in press
12. Irizarry, M., J. R. Wilson, and J. Trevino (2001), A flexible simulation tool for manufacturing-cell design, II: response surface analysis and case study. *IIE Transactions*, 33, pp. 837-846
13. Ivanescu, C., W. Bertrand, J. Fransoo, and J.P.C. Kleijnen (2006), Bootstrapping to solve the limited data problem in production control: an application in batch processing industries. *Journal of the Operational Research Society*, 57, number 1, pp. 2-9
14. Joshi, S., H.D. Sherali, and J.D. Tew (1998), An enhanced response surface methodology (RSM) algorithm using gradient deflection and second-order search strategies. *Computers and Operations Research*, 25, no. 7/8, pp. 531-541
15. Khuri, A. I. (1996), Multiresponse surface methodology. *Handbook of Statistics, volume 13*, edited by S. Ghosh and C. R. Rao, Elsevier, Amsterdam
16. Kleijnen, J.P.C. (1993), Simulation and optimization in production planning: a case study. *Decision Support Systems*, 9, pp. 269-280
17. Kleijnen, J.P.C. (1998), Experimental design for sensitivity analysis, optimization, and validation of simulation models. *Handbook of simulation*, edited by J. Banks, Wiley, New York, pp. 173-223

18. Kleijnen, J.P.C. (2006), White noise assumptions revisited: Regression metamodels and experimental designs for simulation practice. *Proceedings of the 2006 Winter Simulation Conference*, edited by L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto (in press)
19. Kleijnen, J.P.C. (2007), *DASE: Design and analysis of simulation experiments.* Springer Science + Business Media
20. Kleijnen, J.P.C., D. den Hertog and E. Angün (2004), Response surface methodology's steepest ascent and step size revisited. *European Journal of Operational Research*, 159, pp. 121-131
21. Kleijnen, J.P.C., D. den Hertog and E. Angün (2006), Response surface methodology's steepest ascent and step size revisited: correction. *European Journal of Operational Research*, 170, pp. 664-666
22. Kleijnen, J.P.C. and J. Wan (2006), Optimization of simulated systems: OptQuest and alternatives. Working Paper, Tilburg University, Tilburg, Netherlands
23. Khuri, A. I. and J. A. Cornell (1996), *Response surfaces: design and analysis, second edition.* Marcel Dekker, New York
24. Law, A.M. and W.D. Kelton (2000), *Simulation modeling and analysis; third edition.* McGraw-Hill, Boston
25. Myers, R.H. and D.C. Montgomery (2002), *Response surface methodology: process and product optimization using designed experiments; second edition.* Wiley, New York
26. Myers, R.H., A.I. Khuri, and W.H. Carter (1989), Response surface methodology: 1966-1988. *Technometrics*, 31, no., 2, pp. 137-157
27. Oden, J.T., Chair (2006), *Revolutionizing engineering science through simulation.* National Science Foundation (NSF), Blue Ribbon Panel on Simulation-Based Engineering Science
28. Olivi, L. (1984), editor, *Response surface methodology; handbook for nuclear reactor safety.* EUR 9600, Commission of the European Communities, Joint Research Centre, Ispra, Italy
29. Rosen, S.C., C.M. Harmonosky, and M.T. Traband (2006), A simulation optimization method that considers uncertainty and multiple performance measures. *European Journal of Operational Research*, in press
30. Safizadeh, M.H. and R. Signorile (1994), Optimization of simulation via quasi-Newton methods. *ORSA Journal on Computing*, 6, no. 4, pp. 398-408
31. Simpson, T. W., Booker, A. J., Ghosh, D., Giunta, A. A., Koch, P. N., and Yang, R.-J. (2004), Approximation methods in multidisciplinary analysis and optimization: a panel discussion. *Structural and Multidisciplinary Optimization*, 27, no. 5, pp. 302-313
32. Spall, J.C. (2003), *Introduction to stochastic search and optimization; estimation, simulation, and control.* Wiley, New York
33. Stinstra, E.D., H.P. Stehouwer, and J. van der Heijden (2003), Collaborative tube design optimization: an integral meta-modeling approach. *Proceedings of the 5th ISSMO Conference on Engineering Design Optimization*
34. Yang, T. and L. Tseng (2002), Solving a multi-objective simulation model using a hybrid response surface method and lexicographical goal programming approach: a case study on integrated circuit ink-marking machines. *Journal of the Operational Research Society*, 53, no. 2, pp. 211-221